# HLA OMT Data Interchange Format (DIF) v1.1

Document Number:{**0297-101**}
Release Version:{**FINAL**}
Revision: **1**
Author: **Peter R. Valentine**
Created: **21-Feb-97**
Last Updated: **25-Apr-97 11:19 AM**

Prepared by:
**C4I Analysis and Database Branch**
**C4I Test Division**
**ELECTRONIC PROVING GROUND**
**Fort Huachuca, AZ 85613-7110**

## Table of Contents

## 1. Background

The High Level Architecture Object Model Template(HLA OMT) Data Interchange Format (DIF) is a standard file exchange format used to store and transfer HLA Federation Object Models (FOMs) and Simulation Object Models (SOMs) between FOM/SOM Builders. The DIF is built upon a common meta-model which represents the information needed to represent and manage object models created using the HLA OMT standard.

## 2. BNF Notation of the DIF

In order to ensure that there is no ambiguity in the definition of the DIF, we chose to formally define the DIF in terms of Backus Naur Format (BNF). BNF is a formalism used to define programming language syntax. Attributed to John Backus and Peter Naur it was invented to describe the syntax of Algol 60 in an unambigious manner. For the purposes of this document we will be using Extended BNF (EBNF) which includes some additional constructs to handle iteration and alternation as described below:

### 2.1 BNF Notation Conventions

- Certain symbols within the BNF have special meanings, these are called *meta-symbols* and they are used to structure the BNF. Double Quotes, Angle Brackets, Braces, etc. are meta-symbols within BNF, and their definition and use will be given below.
- Words inside double quotes ("word") represent literal words themselves (these are called *terminals*). In addition, terminals will also be highlighted using **boldfaced** text. An example of a terminal is `"`**`HLA-OMT`**`"`.
- Words contained within angle brackets '<>' represent semantic categories (i.e. *non-terminals*) which must be resolved by reading their definition elsewhere in the BNF. An example of a non-terminal is `<NameCharacter>`.
- The BNF used in this document adds a special case of non-terminal which is denoted by double brackets '<<>>' rather than single angle brackets. This special case non-terminal is a reference to the DIF Meta-Model, and further details about the non-terminal, including its definition can be found in the data model glossary.
- A production rule is a statement of the definition of a non-terminal. It is designated by the production meta-symbol '::=' which assigns the definition to the right hand side (RHS) of the production to the non-terminal on the left hand side(LHS) of the production symbol. The LHS must always consist of a single non-terminal, while the RHS can consist of any combination of terminals and non-terminals. An example of a production rule is:
```
        <Integer> ::= 0..65536;
```
  Which defines the non-terminal `<Integer>` to be a number between 0 and 65536.
- Optional Items are enclosed by square bracket meta-symbols '[' and ']'. Square brackets indicate the item exists either zero or one time, that is to say, it may or may not exist. An example of an optional item is `[<Sponsor>]` which indicates the Sponsor item may or may not be present in the DIF.
- Repetition (zero or more) is performed by the Curly Brace meta-symbols '{' and '}'. Curly braces indicate that there may be zero, one, or more sequential instances of the item. An example of curly braces is `{<Version>}` which indicates there may be zero, one or more versions present in the DIF. In basic BNF iteration is described using left or right recursion, extended BNF uses the curly brace meta-symbol to make reading the BNF easier. An example of how the curly brace is used to replace recursion is given below:

  The Rule:
```
         <ident> ::= <Letter> { <Letter> | <Digit> }
```
  is the same as the recursive rule:
```
        <ident> ::=  <Letter> | <ident> [<Letter> | <Digit>]
```
- The double period .. used within a Literal is a shortcut notation for denoting the set of ASCII characters between the characters to either side of them. An example of this is `"a..z"` which denotes the set of lowercase letters between 'a' and 'z' inclusive.
- All BNF Statements are terminated by a semicolon ;

- In order to represent unprintable or reserved characters within the BNF and the DIF code itself, we use the C language convention of using the backslash as the escape character within literals with the following set of legal escaped literals:

| Escape Character | Definition |
|---|---|
| `'\n'` | newline |
| `'\r'` | return |
| `'\''` | Single Quote |
| `'\"'` | Double Quote |
| `'\\'` | Backslash |
| `'\t'` | Tab |
| `'\b'` | Backspace |
| `'\f'` | FormFeed |
| `'\xxx'` | Any character where xxx is its Octal representation |

## 2.2 Basic BNF Constructs

The following are a set of basic BNF constructs referenced in the main body of the DIF BNF, they are defined separately to make the main body more readable.

```
<Integer> ::= 0..65536;

<NameString> ::= <Letter> {<Letter> | <Number> | "." | "_"};

<Letter> ::= "a..z" | "A..Z";
<Digit> ::= "0..9";
<NameCharacter> ::= <Letter> | "_" | "." | <Digit>;

<TextString> ::= <TextString> <TextChar> | <TextChar>;
<TextChar> ::= " "|"!"|"#..\'"|"*"|"+"|"-"|"."|"/"|"0..9"|":..@"|"A..Z"|"[..`"|"a..z"|"{..~";

<Date> ::= <Month> "/" <Day> "/" <Year>;
<Month> ::= ("01"|"02"|"03"|"04"|"05"|"06"|"07"|"08"|"09"|"10"|"11"|"12");
<Day> ::= (("0"|"1"|"2")<Digit>)|("30"|"31");
<Year> ::= <Digit><Digit><Digit><Digit>;

<NoteRef> ::= <OpenBracket> <RefList> <CloseBracket>;
<OpenBracket> :: "[";
<RefList> ::= <RefNumber> {"," <RefNumber>},
<RefNumber> ::= "1..65535";
<CloseBracket> ::= "]";

<DataTypeName> ::= """ <BaseDataType> """ | <<DTP_Name>>;
<BaseDataType> ::= "unsigned short" | "short" | "unsigned long" | "long" | "double" | "float" |
                          "boolean" | "any" | "string" | "char" | "octet";
```

***NOTE**: <NameString> is a YACC Compliant Name which can consist of an Initial letter followed by letters, numbers or the period or underscore. <TextString>  is a string which does not contain double quotes, and can contain embedded "escaped" control codes using the C language conventions noted above.  <DataTypeName> is either a reference to a base data type or the name of a defined <DataType>.  Both base data types and defined data types will be enclosed with quotes.*

# 3. HLA OMT DIF BNF Definition

```
<HLA-OMT_DIF_v1.1> ::= <DIFHeader> {<ObjectModel>};

<DIFHeader> ::= "(DIF " <DIFName> <DIFVerNum> <DIFType>")";
<DIFName> ::= "HLA-OMT";
<DIFVerNum> ::= "v1.1";

<DIFType> ::=  "(TYPE " <DIFTypeName> ")";
<DIFTypeName ::= "Single" | "Multiple" | "Directory";
```

```
<ObjectModel> ::= "(ObjectModel (Name " <<MOD_Name>> ")"
                              "(VersionNumber " <<VER_Number>> ")"
                              ["(Type " <<MOD_Type>> ")"]
                              ["(Class " <<MOD_Class>> ")"]
                              ["(ApplicationDomain " <<MOD_ApplicationDomain>> ")"]
                              ["(Sponsor " <<MOD_Sponsor>> ")"]
                              ["(Developer " <<MOD_Developer>> ")"]
                              ["(POCName " <<MOD_POC_Name>> ")"]
                              ["(POCPhone " <<MOD_POC_Phone>> ")"]
                              ["(POCAddress " <<MOD_POC_Address>> ")"]
                              ["(POCEmail " <<MOD_POC_Email>> ")"]
                              ["(Security " <<MOD_SecurityCharacteristics>> ")"]
                              ["(VVAHistory " <<MOD_VVAHistory>> ")"]
                              ["(ToolUse " <<MOD_ToolUse>> ")"]
                              ["(DocReferences " <<MOD_DocReferences>> ")"]
                              ["(OtherInfo " <<MOD_OtherInfo>> ")"]
                              ["(Purpose " <<MOD_Purpose>> ")"]
                              ["(FedParticipants " <<MOD_FedParticipants>> ")"]
                              ["(KeyFeatures " <<MOD_KeyFeatures>> ")"]
                              ["(CompCapacity " <<MOD_ComputationalCapacity>> ")"]
                              ["(ConfReq " <<MOD_ConfigurationReq>> ")"]
                              ["(ProgLanguage " <<MOD_ProgrammingLanguages>> ")"]
                              ["(TimeManagment " <<MOD_TimeManagement>> ")"]
                              ["(ModificationDate " <<VER_ModificationDate>> ")"]
                              {<OMTComponent>} ")";

<<MOD_Name>> ::= """ <TextString> """;
<<VER_Number>> ::= """ <TextString> """;
<<MOD_Type>> ::= "FOM" | "SOM" | "OTHER";
<<MOD_Class>> ::= "Virtual" | "Live" | "Constructive" | "Hybrid";
<<MOD_ApplicationDomain>> ::= """ <TextString> """;
<<MOD_Sponsor>> ::= """ <TextString> """;
<<MOD_Developer>> ::= """ <TextString> """;
<<MOD_POC_Name>> ::= """ <TextString> """;
<<MOD_POC_Phone>> ::= """ <TextString> """;
<<MOD_POC_Address>> ::= """ <TextString> """;
<<MOD_POC_Email>> ::= """ <TextString> """;
<<MOD_SecurityCharacteristics>> ::= """ <TextString> """;
<<MOD_VVAHistory>> ::= """ <TextString> """;
<<MOD_ToolUse>> ::= """ <TextString> """;
<<MOD_DocReferences>> ::= """ <TextString> """;
<<MOD_OtherInfo>> ::= """ <TextString> """;
<<MOD_Purpose>> ::= """ <TextString> """;
<<MOD_FedParticipants>> ::= """ <TextString> """;
<<MOD_KeyFeatures>> ::= """ <TextString> """;
<<MOD_ComputationalCapacity>> ::= """ <TextString> """;
<<MOD_ ConfigurationReq>> ::= """ <TextString> """;
<<MOD_ProgrammingLanguages>> ::= """ <TextString> """;
<<MOD_TimeManagement>> ::= """ <TextString> """;
<<VER_ModificationDate>> ::= <Date>;

<OMTComponent> ::= <Class> | <Interaction> | <Association> | < DataType> | <Note>;

<Class> ::= "(Class (Name " <<CLS_Name>> [<NoteRef>] ")"
                   ["(PSCapbilities " <<CLS_PSCapbilities>> ")"]
                   ["(Description " <<CLS_Description>> ")"]
                   {<ClassComponent>} ")";
<<CLS_PSCapbilities>> ::= "P" | "S" | "PS" | "N";
<<CLS_Name>> ::= """ <NameString> """;
<<CLS_Description>> ::= """ <TextString> """;

<ClassComponent> ::= <Attribute> | <Component> | < SuperClass>;

<Attribute> ::= "(Attribute (Name " <<ATT_Name>> [<NoteRef>] ")"
                          ["(DataType " <<ATT_DataType>> [<NoteRef>] ")"]
                          ["(Cardinality " <<ATT_Cardinality>> [<NoteRef>] ")"]
                          ["(Units " <<ATT_Units>> [<NoteRef>] ")"]
                          ["(Resolution " <<ATT_Resolution>> [<NoteRef>] ")"]
                          ["(Accuracy " <<ATT_Accuracy>> [<NoteRef>] ")"]
                          ["(AccuracyCondition " <<ATT_AccuracyCondition>> [<NoteRef>] ")"]
                          ["(UpdateType " <<ATT_UpdateType>> [<NoteRef>] ")"]
                          ["(UpdateCondition " <<ATT_UpdateCondition>> [<NoteRef>] ")"]
```

```
                            ["(TransferAccept " <<ATT_TransferAccept>> [<NoteRef>] ")"]
                            ["(UpdateReflect " <<ATT_UpdateReflect>> [<NoteRef>] ")"]
                            ["(Description " <<ATT_Description>> ")"] ")";

<<ATT_Name>> ::= """ <NameString> """;
<<ATT_DataType>> ::= <DataTypeName>;
<<ATT_Cardinality>> ::= """ <TextString> """;
<<ATT_Units>> ::= """ <TextString> """;
<<ATT_Resolution>> ::= """ <TextString> """;
<<ATT_Accuracy>> ::= """ <TextString> """;
<<ATT_AccuracyCondition>> ::= """ <TextString> """;
<<ATT_UpdateType>> ::= "Static" | "Periodic" | "Conditional";
<<ATT_UpdateCondition>> ::= """ <TextString> """;
<<ATT_TransferAccept>> ::= "T" | "A" | "TA" | "N";
<<ATT_UpdateReflect>> ::= "U" | "R" | "UR";
<<ATT_Description>> ::= """ <TextString> """;

<Component> ::= "(Component (ClassName " <<CLS_Name>> [<NoteRef>] ")"
                            "(Number " <<CMP_Number>> ")"
                             ")";

<<CMP_Number>> ::= """ <TextString> """;

<SuperClass> ::= "(SuperClass " <<CLS_Name>> ")";

<Interaction> ::= "(Interaction (Name " <<INT_Name>> [<NoteRef>] ")"
                   "(ISRType " <<INT_ISRType>> [<NoteRef>] ")"
                   ["(Description " <<INT_Description>> ")"]
                   {<InteractionComponent>} ")";
<<INT_Name>> ::= """ <NameString> """;
<<INT_ISRType>> ::= "I" | "S" | "R" | "IS" | "IR";
<<INT_Description>> ::= """ <TextString> """;

<InteractionComponent> ::= <IntMember> | <IntParticipatingClass> | <Parameter>;

<IntMember> ::= "(InteractionMember " <<IMB_Child>> ")";
<<IMB_Child>> ::= """ <NameString> """;

<IntParticipatingClass> ::= ("ParticipatingClass (ClassName " <<CLS_Name>> [<NoteRef>] ")"
                                                "(ParticipantType "<<PCA_Type>> ")"
                                                {<AffectedAttribute>} ")";
<<PCA_TYPE>> ::= "Initiating" | "Receiving";

<AffectedAttribute> ::= "(AffectedAttribute (AttributeName " <<ATT_Name>> [<NoteRef>] ")"
                                            ["(Comment " <<AFA_Comment>> ")"] ")";

<<AFA_Comment>> ::= """ <TextString> """;

<Parameter> ::= "(Parameter (Name " <<PRM_Name>> [<NoteRef>] ")"
                                    "(Order " <<PRM_Order>> ")"
                                    ["(DataType " <<PRM_DataType>> [<NoteRef>] ")"]
                                    ["(Cardinality " <<PRM_Cardinality>> [<NoteRef>] ")"]
                                    ["(Units " <<PRM_Units>> ")"]
                                    ["(Resolution " <<PRM_Resolution>> [<NoteRef>] ")"]
                                    ["(Accuracy " <<PRM_Accuracy>> [<NoteRef>] ")"]
                                    ["(AccuracyCondition " <<PRM_AccuracyCondition>> [<NoteRef>]
                            ")"]
                                    ["(Description " <<PRM_Description>> ")"] ")";
<<PRM_Name>> ::= """ <NameString> """;
<<PRM_Order>> ::= <Integer>;
<<PRM_DataType>> ::= <DataTypeName>;
<<PRM_Cardinality>> ::= """ <TextString> """;
<<PRM_Units>> ::= """ <TextString> """;
<<PRM_Resolution>> ::= """ <TextString> """;
<<PRM_Accuracy>> ::= """ <TextString> """;
<<PRM_AccuracyCondition>> ::= """ <TextString> """;
<<PRM_Description>> ::= """ <TextString> """;

<Association> ::=  "(Association (AssociationName " <<AST_Name>> [<NoteRef>] ")"
                              ["(Description " <<ASO_Description>> ")"]
                              {<AssociationMember>} ")";
<<AST_Name>> ::= """ <TextString> """;
```

```
<<ASO_Description>> ::= """ <TextString> """;

<AssociationMember> ::= "(AssociationMember (ClassName " <<CLS_Name>> [<NoteRef>] ")"
                                          ["(Cardinality " <<AMB_Cardinality>> ")"]
                                          ["(Role " <<AMB_Role>>  ")"]
                                          ["(MemberType " <<AMB_MemberType>> ")"] ")";
<<AMB_Cardinality>> := """ <TextString> """;
<<AMB_Role>> ::= """ <TextString> """;
<<AMB_MemberType>> := "Primary" | "Secondary";

<DataType> ::= <EnumeratedDataType> | <ComplexDataType>;

<EnumeratedDataType> ::= "(EnumeratedDataType (Name " <<DTP_Name>> [<NoteRef>] ")"
                          ["(Description " <<DTP_Description>> ")"]
                          <Enumeration> {<Enumeration>} ")";
<<DTP_Name>> ::= """ <NameString> """;
<<DTP_Description>> ::= """ <TextString> """;
<Enumeration> ::= "(Enumeration (Enumerator " <<ENM_Enumerator>> [<NoteRef>] ")"
                              "(Representation " <<ENM_Representation>> [<NoteRef>] ")"
                   ")";
<<ENM_Enumerator>> ::= """ <NameString> """;
<<ENM_Representation>> ::= <Integer>;

<ComplexDataType> ::= "(ComplexDataType (Name " <<DTP_Name>> [<NoteRef>] ")"
                          ["(Description " <<DTP_Description>> ")"]
                          <ComplexComponent> {<ComplexComponent>} ")";

<ComplexComponent ::= "(ComplexComponent (FieldName " <<CCP_FieldName>> [<NoteRef>] ")"
                                     ["(DataType " <<CCP_DataType>> [<NoteRef>] ")"]
                                     ["(Cardinality " <<CCP_Cardinality>> [<NoteRef>] ")"]
                                     ["(Units " <<CCP_Units>> [<NoteRef>] ")"]
                                     ["(Resolution " <<CCP_Resolution>> [<NoteRef>] ")"]
                                     ["(Accuracy " <<CCP_Accuracy>> [<NoteRef>] ")"]
                                     ["(AccuracyCondition " <<CCP_AccuracyCondition>> [<NoteRef>]
")"] ")";
<<CCP_FieldName>> ::= """ <NameString> """;
<<CCP_DataType>> ::= <DataTypeName>;
<<CCP_Cardinality>> ::= """ <TextString> """;
<<CCP_Units>> ::= """ <TextString> """;
<<CCP_Resolution>> ::= """ <TextString> """;
<<CCP_Accuracy>> ::= """ <TextString> """;
<<CCP_AccuracyCondition>> ::= """ <TextString> """;

<Note> ::= "(Note (NoteNumber " <<NC_NoteNumber>> ")"
                 "(NoteText " <<NC_NoteText>>) ")";
<<NC_NoteNumber>> ::= <RefNumber>;
<<NC_NoteText>> ::= """ <TextString> """;
```

## 4. DIF Representation

This Data Interchange Format (DIF) has been structured as a stream of Object Model meta-data, and does not specify the specific physical representation or transport media used to exchange this meta-data. One possible representation and transport method is to use a simple ASCII File for the exchange of HLA OMT meta-data. The initial prototype Repository and FOM Development tools will use this approach for the interchange of FOMs and SOMs, but this standard does not restrict the interchange to this single approach.

## 5. Types of DIF Instances

The BNF above defines a `<DIFType>` non-terminal whose purpose is to categorize an instance of the DIF into one of three types:

**Single**: A single model and version instance, containing the meta-data of a one version of a single model. This is expected to be the most common use of the DIF, and is the standard unit of exchange between FOM Development Tools and Repositories.

**Multiple**: This keyword indicates a DIF instance where multiple versions and/or models can be included. This type of instance would be used to archive or backup a repository of OMT object models, or as a mechanism for exchanging sets of model/versions between project specific repositories and public repositories.

**Directory**: This keyword indicates a special type of DIF instance, one which is the response to the query of an object model repository for a "directory-listing" of the contents of the repository. This instance would contain only the Model and Version "clauses" which describe the Model and Version meta-data, but would not include the actual model content (i.e. the Classes, Interactions, data types, etc.). This provides the FOM Development Tool with the ability to present their user with a list of the contents of the repository, and the necessary meta-data with which to extract the selected model/version.

## 6. DIF Meta-Data Consistency

When creating a DIF file, it is important that the DIF constructs be consistent. There are definite and well defined relationships between the components of an object model (see the OMT Meta-Model). These relationships must be enforced in order to have a consistent and properly defined DIF. Automated tools must support these relationships in order to produce a compliant DIF instance. An example of consistency is in the definition of affected attributes. If an interaction in the DIF contains an affected attribute `(AffectedAttribute (AttributeName "Att1")` then that attribute must be defined for a Class with an `(Attribute (Name "Att1"))` clause, otherwise an Attribute has been referred to which has not been defined. A number of OMT constructs (entities in the meta-model) have references whose consistency must be maintained:

| Construct | References |
|---|---|
| AffectedAttribute | Attribute |
| AssociationMember | Class |
| Attribute | DataType |
| ComplexComponent | DataType |
| Components | Class |
| InteractionMembers | Interaction |
| Parameter | DataType |
| ParticipatingClass | Class |
| Superclass | Class |

## 7. Sample DIF

The following is a sample DIF File. This sample was built from the examples in the OMT v1.0 Specification and Extensions documents. Where additional data was needed to fully exercise all aspects of the DIF structure, it was added.

```
(DIF HLA-OMT v1.1 (TYPE Single))
(ObjectModel (Name "BNF Test Model")
     (VersionNumber "v1.1")
     (Type FOM)
     (Class Virtual)
     (ApplicationDomain "Training")
     (Sponsor "JDBE")
     (Developer "JDBE")
     (POCName "Pete Valentine")
     (POCPhone "(520) 538-4976")
     (POCAddress "Fort Huachuca, AZ")
     (POCEmail "valentinep@huachuca-emh17.army.mil")
     (Security "Unclassified")
     (VVAHistory "N/A")
     (ToolUse "Joint Headquarters Staff Training")
     (DocReferences "0396-108")
     (OtherInfo "HLA/OMT Doc Version 1.0")
     (Purpose "HLA Federation Prototyping")
     (FedParticipants "TextString")
     (KeyFeatures "Mix of event driven and time driven federates.")
     (CompCapacity "Pentium, PCI/ESIA")
     (ConfReq "Pentium, Windows95")
     (ProgLanguage "C++")
     (TimeManagement "Update Rate")
     (ModificationDate 5/6/96)
     (Class (Name "Appetizer")
       (PSCapabilities S)
       (Description "A portion of food or drink served before a meal.")
          (SuperClass "Food")
     )
     (Class (Name "Beef")
       (PSCapabilities PS)
       (Description "Flesh of a cow for use as meat.")
          (SuperClass "Main_Course")
     )
     (Class (Name "Beef_Barley")
       (PSCapabilities PS)
       (Description "A thin beef-flavored soup.")
          (SuperClass "Soup")
     )
     (Class (Name "Bill")
       (PSCapabilities PS)
       (Description "Statement of money owed.")
     )
     (Class (Name "Cake")
       (PSCapabilities PS)
       (Description "A sweet, baked, breadlike food.")
          (SuperClass "Dessert")
     )
     (Class (Name "Cashier")
       (PSCapabilities PS)
       (Description "The person who collects payments for purchases.")
       (Attribute (Name "Accounts_Receivable")
           (DataType "float")
       )
       (Attribute (Name "Cash_Balance")
           (DataType "float")
       )
       (Attribute (Name "Daily_Receipts")
           (DataType "float")
       )
          (SuperClass "Employee")
     )
     (Class (Name "Chicken")
       (PSCapabilities PS)
       (Description "The flesh of domestic fowl.")
          (SuperClass "Main_Course")
     )
     (Class (Name "Chocolate")
       (PSCapabilities PS)
       (Description "A preparation of the seeds of cacao.")
          (SuperClass "Ice_Cream")
     )
     (Class (Name "Clam_Chowder")
       (PSCapabilities S)
       (Description "Thick soup made of clams or fish and vegetables.")
          (SuperClass "Soup")
     )
     (Class (Name "Coffee")
       (PSCapabilities PS)
       (Description "Beverage derived from coffee beans.")
          (SuperClass "Drink")
     )
     (Class (Name "Cola")
       (PSCapabilities PS)
       (Description "A specific soda flavor made from the syrup of the coca plant or from the
seeds of kola nuts.")
          (SuperClass "Soda")
     )
     (Class (Name "Cook")
       (PSCapabilities PS)
       (Description "Person who prepares the meal.")
       (Attribute (Name "Orders_Pending")
           (DataType "short")       )
          (SuperClass "Employee")
     )
     (Class (Name "Customer")
       (PSCapabilities PS)
       (Description "Patron who wishes to eat a meal.")
       (Attribute (Name "Cash_in_Wallet")
           (DataType "float")
       )
       (Attribute (Name "Satisfaction")
           (DataType "boolean")
       )
     )
     (Class (Name "Dessert")
       (PSCapabilities S)
       (Description "Final course of a meal.")
          (SuperClass "Food")
     )
     (Class (Name "Dishwasher")
       (PSCapabilities PS)
       (Description "Person who cleans plates, pots, or utensils.")
          (SuperClass "Employee")
     )
     (Class (Name "Drink")
       (PSCapabilities S)
       (Description "Liquid that is taken into the mouth and swallowed.")
       (Component (ClassName "Water")
       )
          (SuperClass "Food")
     )
     (Class (Name "Employee")
       (PSCapabilities S)
       (Description "A person working for another person or business.")
       (Attribute (Name "Home_Address")
          (DataType "Address_Type")
          (Cardinality "1")
          (Units "N/A")
          (Accuracy "perfect")
          (AccuracyCondition "always")
          (UpdateType Conditional)
          (UpdateCondition "Employee Request")
          (TransferAccept TA)
          (UpdateReflect UR)
       )
       (Attribute (Name "Home_Number")
          (DataType "string")
          (Cardinality "1")
          (Units "N/A")
          (Accuracy "perfect")
          (AccuracyCondition "always")
          (UpdateType Conditional)
          (UpdateCondition "Employee Request")
          (TransferAccept TA)
          (UpdateReflect UR)
       )
       (Attribute (Name "Pay_Rate")
          (DataType "float")
          (Cardinality "1")
          (Units "Cents/Hour")
          (Resolution "1")
          (Accuracy "perfect")
          (AccuracyCondition "always")
          (UpdateType Conditional)
          (UpdateCondition "Merit Increase" [1])
          (TransferAccept TA)
          (UpdateReflect UR)
       )
       (Attribute (Name "Satisfaction")
           (DataType "boolean")
       )
       (Attribute (Name "Years_of_Service")
          (DataType "short")
          (Cardinality "1")
          (Units "Years")
          (Resolution "1")
          (Accuracy "1")
          (AccuracyCondition "always")
          (UpdateType Periodic)
          (UpdateCondition "1/year, on Anniversary")
          (TransferAccept TA)
          (UpdateReflect UR)
       )
     )
     (Class (Name "Fish")
       (PSCapabilities PS)
       (Description "Completely aquatic, cold-blooded vertebrates having gills and fins.")
          (SuperClass "Seafood")
     )
     (Class (Name "Food" [2])
       (PSCapabilities S)
       (Description "Nourishing substance that is eaten.")
     )
     (Class (Name "Greeter")
       (PSCapabilities PS)
       (Description "A person who welcomes customers to a business establishment.")
          (SuperClass "Employee")
     )
     (Class (Name "Ice_Cream")
       (PSCapabilities S)
       (Description "A frozen food made of cream and variously flavored.")
          (SuperClass "Dessert")
     )
     (Class (Name "Lobster")
       (PSCapabilities PS)
       (Description "A marine, decapod crustacean.")
          (SuperClass "Seafood")
     )
     (Class (Name "Main_Course")
       (PSCapabilities S)
       (Description "The principal dish of a meal.")
          (SuperClass "Food")
     )
     (Class (Name "Manhattan")
       (PSCapabilities PS)
       (Description "Specific type of Clam Chowder made with salt pork and tomatoes.")
          (SuperClass "Clam_Chowder")
     )
     (Class (Name "Nachos")
       (PSCapabilities PS)
       (Description "A thin corn chip.")
```

```
        (SuperClass "Appetizer")
   )
   (Class (Name "New_England")
     (PSCapabilities PS)
     (Description "Specific type of Clam_Chowder made with onions and potatoes.")
        (SuperClass "Clam_Chowder")
   )
   (Class (Name "Orange")
     (PSCapabilities PS)
     (Description "A soda flavor derived from a edible citrus fruit.")
        (SuperClass "Soda")
   )
   (Class (Name "Order")
     (PSCapabilities PS)
     (Description "A specific meal a customer wants to purchase.")
   )
   (Class (Name "Pasta")
     (PSCapabilities PS)
     (Description "Various flour and egg food preparations.")
        (SuperClass "Main_Course")
   )
   (Class (Name "Pepperoni")
     (Description "A Pizza topped with a highly seasoned sausage.")
        (SuperClass "Pizza")
   )
   (Class (Name "Pizza")
     (Description "A flat, open-faced pie covered with cheese, sauce, and other garnishments.")
        (SuperClass "Appetizer")
        (SuperClass "Main_Course")
   )
   (Class (Name "Plain")
     (Description "A Pizza with just sauce and cheese.")
        (SuperClass "Pizza")
   )
   (Class (Name "Root_Beer")
     (PSCapabilities PS)
     (Description "Soft drink flavored with the extracted juices of roots, barks, and herbs.")
        (SuperClass "Soda")
   )
   (Class (Name "Seafood")
     (PSCapabilities S)
     (Description "Any salt-water fish or shellfish used for food.")
        (SuperClass "Main_Course")
   )
   (Class (Name "Shrimp")
     (PSCapabilities PS)
     (Description "Small, long-tailed crustacean.")
        (SuperClass "Seafood")
   )
   (Class (Name "Soda")
     (PSCapabilities S)
     (Description "Carbonated beverage.")
        (SuperClass "Drink")
   )
   (Class (Name "Soup")
     (PSCapabilities S)
     (Description "Liquid food made by boiling or simmering meat, fish, or vegetables.")
        (SuperClass "Appetizer")
   )
   (Class (Name "Vanilla")
     (PSCapabilities PS)
     (Description "Flavoring extract from the Vanilla plant.")
        (SuperClass "Ice_Cream")
   )
   (Class (Name "Waiter")
     (PSCapabilities PS)
     (Description "The person who waits on a table.")
     (Attribute (Name "Cheerfulness")
        (DataType "short")
        (Cardinality "1")
        (Resolution "1")
        (Accuracy "perfect")
        (AccuracyCondition "always")
        (UpdateType Periodic)
```

```
          (UpdateCondition "Performance Review")
          (TransferAccept TA)
          (UpdateReflect UR)
       )
       (Attribute (Name "Efficiency")
          (DataType "short")
          (Cardinality "1")
          (Resolution "1")
          (Accuracy "perfect")
          (AccuracyCondition "always")
          (UpdateType Periodic)
          (UpdateCondition "Performance Review")
          (TransferAccept TA)
          (UpdateReflect UR)
       )
       (Attribute (Name "State")
          (DataType "Waiter_Tasks")
          (Cardinality "1")
          (UpdateType Conditional)
          (UpdateCondition "Work Flow")
          (TransferAccept TA)
          (UpdateReflect UR)
       )
          (SuperClass "Employee")
     )
     (Class (Name "Water")
       (PSCapabilities PS)
       (Description "A compound of hydrogen and oxygen.")
          (SuperClass "Drink")
     )
     (Interaction (Name "Food_Arrives")
       (Description "The base class that indicates the interaction of the  Food_Arrives process.")
       (InteractionMember "Food_Arrives_at_Waiter")
       (InteractionMember "Food_Arrives_at_Customer")
     )
     (Interaction (Name "Food_Arrives_at_Customer")
       (ISRType IR)
       (Description "Indicates food has passed from waiter to customer.")
       (ParticipatingClass (ClassName "Waiter")
          (ParticipantType Initiating)
          (AffectedAttribute (AttributeName "State"))
       )
       (ParticipatingClass (ClassName "Customer")
          (ParticipantType Receiving)
          (AffectedAttribute (AttributeName "Satisfaction")
          )
       )
       (Parameter (Name "Temperature_OK")
          (Order 1)
          (DataType "Temp_Type")
          (Cardinality "1")
       )
       (Parameter (Name "Accuracy_OK")
          (Order 2)
          (DataType "Accur_Type")
          (Cardinality "1")
       )
       (Parameter (Name "Timeliness_OK")
          (Order 3)
          (DataType "boolean")
          (Cardinality "1")
          (Units "N/A")
          (Accuracy "1")
          (AccuracyCondition "always")
          (Resolution "1")
       )
     )
     (Interaction (Name "Food_Arrives_at_Waiter")
       (ISRType IR)
       (Description "Indicates food has passed from cook to waiter.")
       (ParticipatingClass (ClassName "Cook")
          (ParticipantType Initiating)
          (AffectedAttribute (AttributeName "Orders_Pending")
               (Comment "Reduce by 1")
         )
       )
       (ParticipatingClass (ClassName "Waiter")
          (ParticipantType Receiving)
          (AffectedAttribute (AttributeName "State")
       )
       (Parameter (Name "Order_Number")
          (Order 1)
          (DataType "short")
          (Cardinality "1")
          (Accuracy "perfect")
          (AccuracyCondition "always")
       )
       (Parameter (Name "Table_Number")
          (Order 2)
          (DataType "short")
          (Cardinality "1")
          (Accuracy "perfect")
          (AccuracyCondition "always")
       )
     )
     (Interaction (Name "Pay_Bill")
       (Description "The base class that indicates the interaction of the  Pay_Bill process.")
       (InteractionMember "Pay_Bill_by_Credit_Card")
       (InteractionMember "Pay_Bill_by_Cash")
     )
     (Interaction (Name "Pay_Bill_by_Cash")
       (ISRType IR)
       (Description "Indicates method of payment for food received.")
       (ParticipatingClass (ClassName "Customer")
          (ParticipantType Initiating)
          (AffectedAttribute (AttributeName "Cash_in_Wallet"))

       )
       (ParticipatingClass (ClassName "Cashier")
          (ParticipantType Receiving)
          (AffectedAttribute (AttributeName "Daily_Receipts"))
          (AffectedAttribute (AttributeName "Cash_Balance"))
       )
       (Parameter (Name "Bill_Amount")
          (Order 1)
          (DataType "float")
          (Cardinality "1")
          (Units "dollars")
          (Accuracy "perfect")
          (AccuracyCondition "always")
       )
     )
     (Interaction (Name "Pay_Bill_by_Credit_Card")
       (ISRType IR)
       (Description "Indicates method of payment for food received.")
       (ParticipatingClass (ClassName "Customer")
          (ParticipantType Initiating)
       )
       (ParticipatingClass (ClassName "Cashier")
          (ParticipantType Receiving)
          (AffectedAttribute (AttributeName "Daily_Receipts"))
          (AffectedAttribute (AttributeName "Accounts_Receivable"))
       )
       (Parameter (Name "Bill_Amount")
          (Order 1)
          (DataType "float")
          (Cardinality "1")
          (Units "dollars")
          (Accuracy "perfect")
          (AccuracyCondition "always")
       )
       (Parameter (Name "Card_Validity")
          (Order 2)
          (DataType "boolean")
          (Cardinality "1")
          (Accuracy "1")
          (AccuracyCondition "always")
       )
```

```
     )
     (Association (AssociationName "Payment" [3])
       (Description "Entity which shows the relationship between two classes.")
       (AssociationMember (ClassName "Customer")
          (Cardinality "1")
          (Role "Makes the payment.")
          (MemberType Primary)
       )
       (AssociationMember (ClassName "Cashier")
          (Cardinality "1")
          (Role "Receives the payment.")
          (MemberType Primary)
       )
     )
     (EnumeratedDataType (Name "Waiter_Tasks")
       (Description "Creating a user defined data type for each waiter task.")

       (Enumeration (Enumerator "Taking_Order")
          (Representation 1))
       (Enumeration (Enumerator "Serving")
          (Representation 2))
       (Enumeration (Enumerator "Cleaning")
          (Representation 3))
       (Enumeration (Enumerator "Calculating_Bill")
          (Representation 4))
       (Enumeration (Enumerator "Other")
          (Representation 5))
     )
     (ComplexDataType (Name "Address_Type")
       (Description "Defining the complex data type of Address.")

       (ComplexComponent (FieldName "Street")
          (DataType "string")
          (Cardinality "1")
          (Units "N/A")
          (Accuracy "perfect")
          (AccuracyCondition "always")
       )
       (ComplexComponent (FieldName "City")
          (DataType "string")
          (Cardinality "1")
          (Units "N/A")
          (Accuracy "perfect")
          (AccuracyCondition "always")
       )
       (ComplexComponent (FieldName "State")
          (DataType "string")
          (Cardinality "1")
          (Units "N/A")
          (Accuracy "perfect")
          (AccuracyCondition "always")
       )
       (ComplexComponent (FieldName "Zip")
          (DataType "string")
          (Cardinality "1")
          (Units "N/A")
          (Accuracy "perfect")
          (AccuracyCondition "always")
       )
     )
     (ComplexDataType (Name "Temp_Type")
       (Description "Defining the complex data type of Temperature as it applies to a food
item.")

       (ComplexComponent (FieldName "Entree")
          (DataType "boolean")
          (Cardinality "1")
          (Units "N/A")
          (Resolution "1")
          (Accuracy "perfect")
          (AccuracyCondition "always")
       )
       (ComplexComponent (FieldName "Vegie_1")
          (DataType "boolean")
          (Cardinality "1")
          (Units "N/A")
          (Resolution "1")
          (Accuracy "perfect")
          (AccuracyCondition "always")
       )
       (ComplexComponent (FieldName "Vegie_2")
          (DataType "boolean")
          (Cardinality "1")
          (Units "N/A")
          (Resolution "1")
          (Accuracy "perfect")
          (AccuracyCondition "always")
       )
     )
     (ComplexDataType (Name "Accur_Type")
       (Description "Defining the complex data type of Accuracy as it applies to an order of
food.")

       (ComplexComponent (FieldName "Entree")
          (DataType "boolean")
          (Cardinality "1")
          (Units "N/A")
          (Resolution "1")
          (Accuracy "perfect")
          (AccuracyCondition "always")
       )
       (ComplexComponent (FieldName "Vegie_1")
          (DataType "boolean")
          (Cardinality "1")
          (Units "N/A")
          (Resolution "1")
          (Accuracy "perfect")
          (AccuracyCondition "always")
       )
       (ComplexComponent (FieldName "Vegie_2")
          (DataType "boolean")
          (Cardinality "1")
          (Units "N/A")
          (Resolution "1")
          (Accuracy "perfect")
          (AccuracyCondition "always")
       )
     )
     (Note (NoteNumber 1)
           (NoteText "Merit raises are not provided according to any regular time interval, but
are provided on a supervisor\'s recommendation based on evidence of exceptional performance."))
     (Note (NoteNumber 2)
           (NoteText "The Food superclass is categorized into many food group classes."))
     (Note (NoteNumber 3)
           (NoteText "The customer will leave a tip for the waiter if the customer received good
service."))
)
```